Tri-Lab Linux Capacity Clusters (TLCC) at Los Alamos

The TriLab Linux Capacity Cluster (TLCC) systems are HPC Linux Clusters in all three networks available to ASC Computing users. They are Appro International, Inc. clusters identical in design and architecture to the TLCC platforms at both Livermore (LLNL) and Sandia (SNL) National Laboratories. TLCC access is restricted to ASC nuclear weapons research, but open to some non-ASC collaborative work in the Turquoise network. For a synopsis of user information, you can you can download the TLCC Quick Reference Card.

Cluster Names	Lobo, Turing, and Hurricane
Account Information	Lobo, Turing, and Hurricane are ASC Program resources. Approval required, restricted to users requiring access for nuclear weapons research. For more information, see the article on Getting an Account.
Network Location	Lobo is in the Turquoise (Open Collaborative, OCN) Network, Turing is in the Yellow (Unclassified Protected, UPN) Network, and Hurricane is in the Red (Secure Restricted, SRN) Network.
Accessing TLCC (Front End and Compile Nodes)	Use `ssh` to reach a front-end/compile node. For Lobo in the Turquoise: `lo-fe1.lanl.gov` and `lo-fe2.lanl.gov`. In the Yellow network, use `lo-fe3.lanl.gov` for compiling export-controlled source code (but **not** for submitting jobs). Please see the Turquoise link for information on accessing wtrw.lanl.gov to reach these front-end nodes. For Turing in the Yellow network, use `tu-fe1`. For Hurricane in the Red network, `hu-fe1.lanl.gov` and `hu-fe2.lanl.gov`.
Base Architecture	The TLCC clusters are comprised of *Connected Units* (CUs). These CUs contain compute nodes and I/O nodes that employ the Quad-Core AMD Opteron third-generation processor. The nodes of a CU are connected together with a Voltaire Infiniband (IB) 4 x Dual Data Rate (DDR) network in a fat tree topology, using a Grid Director ISR 2012 switch. Each 2xCU cluster at 2.2 GHz has a theoretical peak of ~38.3 teraflops and 8.7 TeraBytes of RAM, and Turing has a theoretical peak of 9.4 teraflops and 2 TB of RAM. • Hurricane has two full CUs at the 2.2 GHz speed plus a third CU with 88 compute nodes that run at 2.3 GHz. • Lobo has two full CUs at 2.2 GHz. • Turing has one partial CU with only 64 nodes running a 2.3 GHz clock.
Cluster Hardware Configuration	Each TLCC cluster contains at least one segment or *connected unit* (CU), the building-block. Each CU has 136 compute nodes for performing parallel computations and 6 I/O nodes to handle data traffic (*Two exceptions*, Turing has only 64 total compute nodes, and Hurricanes third CU, `huc` has 88 compute nodes). Each node has 4 sockets housing the Quad-Core AMD Opteron for a total of 16 cores per node. All sixteen share 32 GB of RAM on each node. This provides a total of 2176 compute-node cores and 4.35 TB of RAM per CU. Thus, a 2 CU cluster such as Lobo contains 4,352 x Opteron 2.2 GHz cores and 8.7 TeraBytes of RAM. As an example, the **theoretical peak** performance of Lobo is

	4352 cores x 4 flops per clock cycle x 2.2 GHz clock cycle, or 38.3 TFlops. Hurricane has the same peak plus an additional [88 nodes x 16 cores/node x 4 flops/clock x 2.3 GHz] 12.9 TFlops for a total of 51.2 TFlops. Note that the AMD processor stores data in a little endian format.
Compute Node Architecture	• A node is comprised of 4 x Quad-Core AMD Opteron model 8354 @ 2.2 GHz or the model 8356 @ 2.3 GHz. The Quad-Core AMD Opteron is a single chip/die containing four cores/CPUs/processors that occupy one of the four sockets in a node. • A compute node has a total of 16 cores, each with 0.5 MB secondary cache. • Each Quad-Core AMD Opteron chip shares a 2MB tertiary (L3) cache between the four cores. • The four Opteron sockets (ie. all 16 cores) share 32 GBytes of RAM on each node. • For more information on the Opteron, see the architectural features.
Operating System	Clustered High Availability Operating System (CHAOS). CHAOS is a Livermore modified version of RedHat Linux.
Home Directory	• For the Lobo cluster in the Turquoise network, `/users/username`, shared among all of Lobo, but NOT shared with the other ICN clusters. Home space is limited to 500MB per user. Alternative: `/usr/projects/projectname/username` provides 30GB of NFS space. • For the Turing cluster in the Yellow network, `/users/username` is shared among all HPC clusters such as Yellowrail and rr-dev. • For the Hurricane cluster in the Red network, `/users/username` is shared among all HPC clusters such as Redtail and Roadrunner.
Scratch Space	Globally accessible `/scratch[2,3]` via the Panasas parallel file system. 800 TB in the Turquoise, 144 TB in the Yellow, 658 TB in the Red. For useful information on efficient I/O, see the article on Best I/O Practices.
Give Directory	The give directory is `/net/givedir/username`
Batch System	Moab Workload Manager (ie. scheduler) with the SLURM resource manager.
Support	Lobo and Hurricane service coverage is in the 5x9 category. Contact ICN Consulting for direct user support.
Training	See the ASCI Training web site on the yellow network (http://int.lanl.gov/projects/asci/training/). Look for classes on *Introduction to HPC at LANL* and *Using Moab at LANL*.

Accessing HPC Platforms in the Turquoise (Open Collaborative) Network

Here are instructions for accessing, logging-in, logging-on, connecting, authenticating to the HPC platforms in our Turquoise (Open Collaborative) network.

1. `ssh wtrw.lanl.gov`

   To reach Turquoise HPC clusters, you will need three things: an approved and established HPC account, your LANL Cryptocard, and `ssh` software on your workstation to get through the ssh proxy firewall system, `wtrw.lanl.gov`. This is true whether you come in from the Unclassified Protected Network (UPN -- yellow) or from anywhere on the internet.

2. ssh *front_end_node*

   From the `wtrw.lanl.gov` proxy server, you will need to log-in to a front-end (aka. compile) node of an HPC cluster. This is the only path available to reach HPC compute nodes.

   The Turquoise front-end nodes are (all with .lanl.gov domain):

Turquoise Front-end / Compile nodes	
Lobo	lo-fe1, lo-fe2
Coyote	cy-c1, cy-c2, cy-c3
Cerrillos	ce-fe1, ce-fe2
Garnet	ga-fe1

3. **SHORTCUT**: You can use the `ssh -t` option to perform the login in a single hop. It will prompt you for a passcode; just use your CRYPTOcard to generate one. Also, this is the most convenient (and possibly only) way to open an X11 window (eg. TotalView) for displaying back to your tty using the `ssh -X` command line option. Sample syntax:

   ```
   ssh -t -X wtrw.lanl.gov ssh lo-fe2.lanl.gov
   ```

IMPORTANT

At four hours of idle time, your login session into the firewall will freeze until you re-authenticate with your LANL Cryptocard. Idle time refers to typing at the keyboard -- data transfers (scp, etc.) do not count against idle time. Also, there will be a 16-hour absolute timeout from the last re-authentication.

NOTES

The ssh client is available for use on all LANL networks. If you attempt to reach a front-end directly from a Microsoft system, you will need to first install an SSH client, which is available for download in the protected (yellow) network at http://esd.lanl.gov. Be sure your ssh client supports protocol 2.

The HPC front-end / compile nodes allow you to compile your applications, launch, monitor, and modify jobs, and also provide the portal to reach compute nodes. From the front-end node, you can use the `llogin` command for interactive access, Moab (via the `msub` command), or LSF (via the `bsub` command) to reach compute nodes and run jobs there.

EXPORT CONTROLLED SOURCE CODE

Some users have export-controlled source code, which is forbidden inside the Turquoise network. Therefore, we provide tiny compile nodes for people to generate object code and executable code in the Yellow (Unclassified Protected) network that they can transfer into the Turquoise. These Yellow network machines are (all with .lanl.gov domain):

Compile Nodes in the Yellow Network	
Lobo	lo-fe3
Coyote	cy-c4
Cerrillos / Garnet	rr-dev

Example

```
[robz:~] rtc% ssh wtrw.lanl.gov
rtc@wtrw.lanl.gov's password:

******************************************************************************
This is a Federal computer system and is the property of the United
```

...(abbreviated)...

```
and conditions of use.   LOG OFF IMMEDIATELY if you do not agree to the
conditions stated in this warning.
******************************************************************************


You are logging in under a restricted shell which allows ssh
connections to hosts throughout the Turquoise Network.

You will be prompted to re-authenticate every four(4) hours, regardless of
the level of activity seen on the connection.  Five(5) minutes prior to
the prompt, you will receive a warning message.  While in the
re-authentication phase, all tunnels will be suspended.  Please be
prepared to re-authenticate every 4 hours.

rtc@wtrw.lanl.gov>
```

```
rtc@wtrw.lanl.gov> ssh lo-fe2
Last login: Tue Oct 28 14:49:46 2008 from wtrw.lanl.gov
lo2-fe> module list
Currently Loaded Modulefiles:
  1) intel-f/10.0.023      2) openmpi-intel/1.2.8
lo2-fe> showstate
cluster state summary for Thu Dec 11 15:15:33


    JobID              S      User    Group Procs   Remaining            StartTime
    ------------------ - --------- -------- ----- -----------  -------------------

(A) 58807              R       afn    users  1024     4:26:18  Thu Dec 11 10:41:51
(B) 58808              R       afn    users    16  1:22:59:10  Thu Dec 11 14:14:43
(C) 58810              R       afn    users    16  1:22:59:41  Thu Dec 11 14:15:14
(D) 58811              R       afn    users    16  1:22:59:41  Thu Dec 11 14:15:14
(E) 58813              R       afn    users    16  1:23:00:43  Thu Dec 11 14:16:16
(F) 58814              R       afn    users    16  1:23:00:43  Thu Dec 11 14:16:16
```

...Truncated...

```
(4) 58845              R       afn    users    16  1:23:04:53  Thu Dec 11 14:20:26
(5) 58846              R       afn    users    16  1:23:04:53  Thu Dec 11 14:20:26
(6) 58847              R       afn    users    16  1:23:04:53  Thu Dec 11 14:20:26

usage summary:  33 active jobs  96 active nodes

              [0][0][0][0][0][0][0][0][0][1][1][1][1][1][1][1]
              [1][2][3][4][5][6][7][8][9][0][1][2][3][4][5][6]

Rack      01: [ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][!]
Rack      02: [ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ]
Rack      03: [ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ]
Rack      04: [ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ]
Rack      05: [ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ]
Rack      06: [ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ]
Rack      07: [ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ]
Rack      08: [A][A][A][A][A][A][A][A][A][A][A][A][A][A][A][A]
Rack      09: [A][A][A][A][A][A][A][A][A][A][A][A][A][A][A][A]
Rack      10: [A][A][A][A][A][A][A][A][A][A][A][A][A][A][A][A]
Rack      11: [A][A][A][A][A][A][A][A][A][A][A][A][A][A][A][A]
Rack      12: [ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ]
Rack      13: [ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ]
Rack      14: [ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ]
Rack      15: [ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ]
Rack      16: [6][5][3][2][1][4][0][Z][Y][X][W][U][T][S][R][V]
Rack      17: [Q][P][O][N][M][L][K][J][I][H][G][F][E][D][C][B]

Key:  [?]:Unknown [*]:Down w/Job [#]:Down [ ]:Idle [@] Busy w/No Job [!] Drained
lo2-fe>
```

Compiling and Executing Applications on TLCC Platforms

In order to compile and run your application on a TLCC platform, you will need to reach a front-end/compile node. Note that you can also compile on a back-end/compute node if you need exclusive use of 16 CPUs and 32 GB of memory. However, that will affect your Moab fairshare history and it is unusual to need such power for compilation. If necessary, also load an MPI modulefile. Here is an example:

1. First, I determine what modulefiles are available (ie. which compilers, libraries, tools, etc.)

```
lo2-fe> module avail

------------------- /usr/share/modules/modulefiles/compiler --------------------
Compilers:
...truncated (a long output to the shell)...
```

2. In this example, load both an Intel compiler and the corresponding MPI library:

```
lo2-fe> module load intel-f openmpi-intel/1.3.2
```

3. Then compile using the MPI compile wrapper:

```
lo2-fe> mpif90 -o lobo_mpitest mpi_test.f90
/opt/Intel/fce/10.0.023/lib/libimf.so: warning: warning: feupdateenv is
not implemented and will always fail
lo2-fe>
```

4. Now the executable is ready to run, so the next step is to obtain an allocation on a compute node. You accomplish this with the Moab command, msub. When executing on a compute node, beware that you must have the same modulefiles loaded there. In this example we ask for an interactive multi-node allocation for 20 minutes:

```
lo2-fe> msub -I -l nodes=2:ppn=16,walltime=20:00
salloc: Job is in held state, pending scheduler release
salloc: Pending job allocation 72960
salloc: job 72960 queued and waiting for resources
salloc: job 72960 has been allocated resources
salloc: Granted job allocation 72960
lob096.localdomain>
```

5. Then check for the same modulefiles and use "mpirun" to execute your parallel code:

```
lob096.localdomain> module list
Currently Loaded Modulefiles:
  1) intel-f/10.0.023      2) openmpi-intel/1.2.8
lob096.localdomain> mpirun -np 32 lobo_mpitest
  Hello World, I am process            1
  Hello World, I am process            2
  Hello World, I am process           17
  Hello World, I am process            3
  Hello World, I am process           18
  Hello World, I am process            4
  Hello World, I am process           19
```

...truncated...

Filesystems and Storage on Open Collaborative (Turquoise) HPC Linux Clusters

Most of the Linux clusters in the Open Collaborative (Turquoise) Network have multiple mounted filesystems that serve different purposes. Here we present a general overview of what you can expect to see. Note that the HPC clusters in the Turquoise network are configured with a different security model than the other networks. Home directories are not shared between platforms, but we do cross-mount the project spaces and the scratch spaces.

Large Temporary Storage -- Spend Your Time Here

usage	temporary, large, parallel I/O
path name	`/scratch1/$USER` and `/scratch2/$USER`
quota	none
backed-up	not backed-up Note: this filesystem is purged automatically on a regular basis
shared	between all Linux clusters in the Turquoise network (ie. Coyote, Lobo, etc.)

The largest mounted filesystems are also temporary, volatile storage. Here we have 800 TB of Panasas filesystems where we expect to see the bulk of user I/O activity, especially large scale parallel I/O. These are global parallel filesystems mounted on all compute nodes and accessible from the front-end systems.

We recommend you archive all important scratch files on a daily basis. None of the scratch files are backed-up, and you are on your own to protect them. The scratch filesystems are **not** intended for long-term storage and can be affected by system outages, reboot, network and interconnect problems, etc. Also, these filesystems are subject to a purge policy. For information on I/O optimization, see the Best Practices guide. Our scratch filesystems are products of Panasas, Inc.

NFS Mounted Filesystems

usage	small, permanent storage
path name	**home**: `/users/`*`username`* **project**: `/usr/projects/`*`projectname`*`/`*`username`*
quota	**space**: 500 MB for home directory **number of files**: 100 K in home directory
backed-up	**home**: `/usr/projects/`*`projectname`*`/`*`username`*`/.snapshot`
shared	**home**: not shared **project**: with all Turquoise clusters (ie. Lobo, Coyote, etc.)

Some important notes regarding NFS-based filesystems:

- Your home directory is unique and is **not** shared with any other HPC cluster. It will appear with this pathname: `/users/username`. It exists only for your dotfiles (.login, .cshrc, etc.) and has a tiny allocation (5MB); thus it is not designed for daily usage or working space. Your home directory is backed-up to the Tivoli Storage Manager (TSM) on a regular basis. Warning: if you place a `cd` command into your `.cshrc` file, this will cause problems with scripts and also with commands such as scp.

- On all nodes, we will mount a minimum set of NFS filesystems. These filesystems are backed-up to TSM on a regular basis, and they are never purged.

- We provide private *workspace* for you to retain small permanent files, logs, source code, to perform small serial I/O operations, etc. The workspace is similar to your home directory, without the dotfiles (.login, .cshrc, etc.). Your workspace is NFS mounted and shared among all Turquoise HPC clusters using a common pathname: `/usr/projects/projectname/username`. The *projectname* portion of your new workspace pathname is based on your membership in a group/project/account (see the `bugroup` command).

- Your workspace might appear to be missing if it is not already mounted on /usr/projects. You can use the `ls` or `cd` command to mount it:

```
pfe1.lanl.gov> pwd
/usr/projects
pfe1.lanl.gov> ls
blkHole  proTher  support
pfe1.lanl.gov> ls fundPhys
carlson
pfe1.lanl.gov> ls
blkHole  fundPhys  proTher  support
pfe1.lanl.gov>
```

- Your project work space is different from your home directory and also different from the project space in the UPN (Yellow Network).

Permanent Long-term Storage

Long-term, archival storage is available on our General Parallel File System (GPFS). It is mounted on special GPFS nodes that are accessible from the Turquoise firewall, `wtrw.lanl.gov`.

Locally Mounted Filesystems

Some HPC Linux Clusters have a small set of locally mounted filesystems. Only one of these are accessible for user applications to test and debug: `/tmp`. This is **not** permanent storage and files may disappear from one application to the next. There is also no guarantee the scheduler will allocate the same set of nodes for a subsequent job. Note that these /tmp filesystems are not built for parallel access from your application. Do not rely on `/tmp` for production computing.

Job Scheduling on TLCC Platforms

On our TLCC clusters, we have established the following set of configurations and parameters for job scheduling with the Moab workload manager. This page is mostly a summary of the TLCC QoS Parameters Plan from X-Division.

For more information on using Moab to manage your jobs, see the Moab home page or the Moab example page. Also, take note of the TLCC Idiosyncrasies, and be sure to print-up a copy of the TLCC Quick Reference Card.

Lobo Cluster	
Time Distribution	• 75% -- Advanced Simulation and Computing (ASC) • 25% -- Institutional Computing (IC) Your jobs will run on compute nodes based on a pre-determined allocation of your project using the Moab fairshare scheduler.
Defaults	2-hour default walltime and one node with 16 physical processor elements (ie. PEs, processor "cores"). Minimum is one node; no oversubscription of jobs.
Maximums	The maximum walltime for any job is 16 hours, regardless of process count. The maximum process count for any single job is 2,144, and for any user is 2,176.

Programming Considerations

We are not yet oversubscribing nodes, and your jobs will allocate a minimum of one node.
Our scheduling algorithm favors smaller PE jobs.
When your job launches and begins to run, it has exclusive access to all 16 cores of each node in its allocation.
Your job can run one process per node with access to all 32 GB of node memory using the `mpirun -npernode` option. This is described in the TLCC Idiosyncrasies page.
Your job can oversubscribe any of its nodes with its own processes or threads.
There are limited PEs in each cluster. See the summary table for specifics.
We are using a queueless scheduler, there will be only one queue (ie. class). There is no need to use the -q option with `msub`.
All batch job submissions run your scripts using the `sh` shell by default. This is different from all other Moab implementations at LANL and will cause problems with modulefiles. For more information see the idiosyncrasies page.

We would like your feedback on this job mix and scheduling parameters as you gain experience with Lobo, Hurricane, and Turing. Send email to consult@lanl.gov.

TLCC Idiosyncrasies

Here we list unique features, discrepancies, configuration differences, and non-standard implementations of software, filesystems, and user environment on the TLCC platforms (Lobo, Hurricane, and Turing). We chose to make the cluster available to users sooner rather than wait for extended software development work to finish later.

- While TLCC platforms use the Moab workload manager (job scheduler, ie. batch system), their underlying resource manager is the Simple Linux Utility for Resource Management, or SLURM. Most of the Moab commands behave as expected, but you may see differences between using Moab on the TLCC platforms vs. Moab on other LANL clusters. For instance, the `bpeek` command does not work at all on TLCC. Lobo users, be sure to see our compilation of Lobo scheduling issues.

- The SLURM scheduler runs your batch scripts under the `sh` shell, but only after first starting a new login shell using your default. This is incompatible with all other platforms using Moab at LANL, and the modulefiles are not configured correctly under the sh/bash shells. You can work around this problem by using one of the following adjustments:

  1. If you prefer to use tcsh, include a new first line in your script: `#!/bin/tcsh`

  2. A different option for tcsh users, explicitly specify your shell within the command line: `msub -S /bin/tcsh` *`script`*

  3. For sh/bash users, include the correct sh/bash setup command for modulefiles within your script: `./usr/share/modules/init/bash`

  4. An alternative for sh/bash users, specifically force a login sh/bash shell by adding a new first line to your script: `#!/bin/bash -l`

- Interactive logins to compute nodes (ie. `msub -I`) force an implicit `-V` option in addition to executing (sourcing) your `.cshrc` file and ignoring your `.login` file. This behavior is the opposite of what most users expect of a new login shell. It means all environment variables from the parent shell on the front-end (from where you issued the `msub` command) get passed-on to the children shells on the compute nodes. This may cause problems if you

  - desire different values for the same environment variables between the front- and back-ends

  - load different modulefiles between front- and back-ends

  - expect your new compute node interactive session to be a login shell

  Our Moab provider, Cluster Resources, Inc., is working on a repair to remove the implicit `-V` option.

  As an alternative, we prefer you use our locally-developed `llogin` command. This provides you a normal interactive shell with a fresh login on a compute node. Currently `llogin` does not provide an X11 tunnel (for Xwindows) to the TLCC compute nodes, but otherwise behaves the same across all our Moab-based clusters.

- X11 tunneling is not straightforward for user applications running on back-end compute nodes. See

instructions here.

- The MPI libraries available are Mvapich (for gcc) and Open MPI. With mvapich, you will need to use `srun` instead of `mpirun` or `mpiexec`. Depending on the version of mvapich, you may have to specify `--mpi=mvapich` on the srun command line. For example:

  ```
  srun -n16 --mpi=mvapich my.exe etc.
  ```

  The `mpirun` command will force all mvapich processes to run on a single remote node, regardless of process count, even if there are hundreds. The `module help mvapich-...` command offers a bit of information on compiling.

- Users can compile on both the front- and back-ends. However, the Pathscale compiler is available only on the front-end nodes. Both the Intel and PGI compilers are available on all nodes, front- and back-ends.

- Users cannot send email from the compute node back-ends. The front-ends are fine for sending email, but the back-ends do not have an external network connection to provide it.

- **Note**: The `msub -l nodes=N:ppn=M` command line option does not limit your application to only M processes per node. If you wish to run a parallel application on Lobo using fewer than 16 processes per node, you can do so by using the `mpirun -npernode` command line option. First, use `msub` to obtain your node allocation, then load an an openmpi modulefile before running the `mpirun` command:

  ```
  rtc@wtrw.lanl.gov> ssh lo-fe1
  Last login: Tue Dec 22 15:36:47 2009 from wtrw.lanl.gov
  ******************************************************************************
                                Notice to Users


  ......truncated.......

  ******************************************************************************
  lo1-fe> msub -I -l nodes=3,walltime=10:00
  salloc: Job is in held state, pending scheduler release
  salloc: Pending job allocation 87707
  salloc: job 87707 queued and waiting for resources
  salloc: job 87707 has been allocated resources
  salloc: Granted job allocation 87707
  lob045.localdomain> mpirun -n 6 -npernode 2 hostname
  lob045.localdomain
  lob045.localdomain
  lob047.localdomain
  lob047.localdomain
  lob046.localdomain
  lob046.localdomain
  lob045.localdomain>
  ```

  This example asks for three nodes with two processes per node. In this way, each process can utilize up to 16 GB of memory.

Xwindows from TLCC -- establishing an X11 connection

To establish an Xwindow (ie. X11) session from your workstation to a TLCC compute node (Lobo, Hurricane, etc.), it is important to do two things:

1. Use an Xwindows client to log-in to a front-end node using the `ssh -X` command line option. (Turquoise users have more typing: connect through the Turquoise firewall to a front-end node using `ssh -X -t wtrw.lanl.gov ssh lo-fe1.lanl.gov`.)

2. Use the `llogin` command.

These two steps are necessary if you wish to debug using Totalview, run a graphics package, use `xterm`, etc. This is all you need to know if you are an experienced user with LANL HPC platforms. Otherwise, read on.

More Details

1. New TLCC users see the TLCC Home Page for front-end node and access information. Lobo users, see instructions described in the Accessing Turquoise page.

2. Connect to a front-end, ensuring you use the `ssh -X` command-line option.

3. Establish a compute-node/back-end allocation on the cluster. In other words, issue an `llogin` command. This requests resources in the form of a Moab *job*, and provides you an interactive shell on a compute node with access to all nodes in your job allocation. (`llogin` is a wrapper for a Moab `msub` command)

4. Of course, you can always use the `msub -V` command from the front-end instead, but you will need to ensure your environment variables and modulefiles are properly instantiated in the shell running on your compute node allocation.

5. At this point, you should be able to issue an xterm command from the shell on the compute-node.

Attaching to an Existing Job

1. If you wish to attach Totalview to a job that is already running on a TLCC compute node, first connect to a front-end, ensuring you use the `ssh -X` command-line syntax. (Turquoise users have more typing: connect through the Turquoise firewall to a front-end node using `ssh -X -t wtrw.lanl.gov ssh lo-fe1.lanl.gov`.)

2. From the front-end node, find out which compute nodes are assigned to your existing, running job. Use the `showq -u your_username` command to obtain your JobID, and then run `checkjob jobid` to see its compute-node list.

3. Then choose the lowest-numbered node and issue an `ssh -X` to that node. From this window, your `xterm` or `totalview` commands should display back to your workstation. Ensure you have the desired `totalview` modulefile loaded.